REMARKS—General

After this amendment, claims 1-3, 5, 16, 18-32 will be pending. Claims 1, 5, 16, 19-21, 23, 25-27, 30-32 have been amended.

Summary of changes to claims

Claim 1: The capability in 1(b)(3) was described as applying to "one or more" interior nodes.

1(4)(i) was amended to further clarify the general concept of extending a sequence of mutually-consecutive tuples, by modifying field(s) representing the sequence length in the existing structure, rather than by using more storage space. This was done to include tuple-sequence representations that do not explicitly recite the length, e.g. representing a sequence by its first and last indexes. (The first and last indexes are capable of representing the length: (length = last – first + 1)).

The former claims 5, 25, and 30 were rewritten in independent form as claim 20, upon the Examiner's suggestion. The limitations of their common base claim have been included. The previous text of claims 5, 25, and 30 has been replaced.

Claim 5 adds a new limitation to the previous claim of representing separate leaves and dictionaries (formerly claim 20): that a plurality of trees share a common dictionary. This limitation was added to further distinguish the present invention from prior art. Also see claims 25 and 31).

Claim 16 combines the use of a gate field with the use of mutually-consecutive tuples (30), and the tree-construction method (20).

Claim 19 combines the step of adding a record subset to a tree, with the step of reconstructing a stored record subset (18) and the use of a gate field (3).

Claim 20 is the new independent claim suggested by the Examiner, reciting a method for constructing a tree with a problem-space search, combined with the previous base-claim limitations. (The problem space was a dependent claim in the former claims 5, 25, and 30). The previous method of claim 20 (distinct leaves and dictionaries) has been rewritten as dependent claims in 5, 25, and 31.)

Claim 21 is a previously presented dependent claim of the tree-construction method (20), (moved here from the former claims, 16, 31, and 32).

Claim 23 had a minor rewording.

Claim 25 adds a new limitation to the previous claim of representing separate leaves and dictionaries (formerly claim 20): that a plurality of trees share a common dictionary. This limitation was added to further distinguish the present invention from prior art. Also see claims 5 and 31).

Claim 26 had a minor rewording.

Claim 27 had minor rewordings.

Claim 30 combines the use of sequences of mutually-consecutive tuples with the tree-construction method (20).

Claim 31 adds a new limitation to the previous claim of representing separate leaves and dictionaries (formerly claim 20): that a plurality of trees share a common dictionary. This limitation was added to further distinguish the present invention from prior art. (Also see claims 5 and 25).

Claim 32 adds value and tuple counts to the tree-construction method (20).

The foregoing amendments are taken in the interest of expediting prosecution and there is no intention of surrendering any range of equivalents to which Applicant would otherwise be entitled in view of the prior art.

By amending the application, the Applicant does not concede that the patent coverage available to them would not extend as far as the original claim. Rather, Applicant reserves the right to file a continuation application to pursue the breadth of the claims as filed. Applicant believes that the Examiner has not made a sufficient showing of inherency of the teachings of the asserted prior art, especially given the lack of teachings in the cited references of the properties that Applicant has recited in his claims.

Further, by the present amendment, it does not follow that the amended claims have become so perfect in their description that no one could devise an equivalent. After amendment, as before, limitations in the ability to describe the present invention in language in the patent claims naturally prevent the Applicant from capturing every nuance of the invention or describing with complete precision the range of its novelty or every possible equivalent. See, Festo Corp. v. Shoketsu Kinzoku Kogyo Kabushiki Co., 62 USPQ2d 1705 (2002). Accordingly, the foregoing amendments are made specifically in the interest of expediting prosecution and there is no intention of surrendering any range of equivalents to which Applicant would otherwise be entitled.

Rejection of claims 1-3, 16, 17, 19-24, and 26-28 under 35 U.S.C. 102(b)

1. Claims 1-3, 16, 17, 19-24, and 26-28 were rejected as being anticipated by Bugajski (US Patent No. 5,592,667).

2. Regarding on claim 1, Applicant respectfully traverses the rejection of claim 1 over Bugajski. Bugajski does not disclose each element of the claimed invention.

Claim 1(b)(3) is said to be disclosed by Bugajski (col. 15, lines 20-25), and claim 1(b)(4)(i) is said to be disclosed by Bugajski (col. 8, lines 11-13).

The Applicant respectfully asserts that claims 1(b)(3) and 1(b)(4)(i) recite methods that are not disclosed by Bugajski.

Claim 1(b)(3) states:

"interior nodes are capable of storing one or more sequences of mutually-consecutive tuples by representing said sequences in a form that uses less storage space than representing said sequences as individual tuples"

This is said to be disclosed by Bugajski (col. 15, lines 20-25)(Claim 1), which states:

"... each non-terminal node of the tree being represented by an associative memory assigning a numerical index value to each unique combination of the index values of the two nodes from which that non-terminal node is derived"

The Applicant respectfully asserts that claim 1(b)(3) recites methods that are not disclosed by Bugajski:

First, claim 1(b)(3) describes a method of optimizing space for the storage of the index-value associations described by Bugajski. In particular, it recites that the tuples (associations) of a particular input sequence type, can be represented in less space than the space required to distinctly represent each individual tuple. Thus the claim recites a form wherein for a tuple sequence of length, n:

SpaceRequired (TupleSequence) < n * SpaceRequired (SingleTuple)

The compressed form in the present invention is shown to use less storage space to store a tuple sequence than storing said sequence as individual tuples, (application, (page 8, paragraphs 2 and 3). This example compares prior art (application, Figure 2) with the present invention (application, Figure 3). Figure 3 (65)) is a single entry that represents the tuple sequence of rows 2 through 10 in Figure 2. Said entry comprises the token, and left and right tuple elements for the first tuple in a tuple sequence, and the sequence length (which could be used to reconstruct said sequence).

Said single entry is shown to require one-ninth the space required to store said tuple sequence as individual tuples, (application, page 8, paragraph 3).

Bugajski-claim 1 describes an associative memory that associates index values to unique combinations of other index values. Said associative memory comprises a set of associations (also known as "tuples", Bugajski, col. 3, lines 30-35). Bugajski-claim 1 does not teach any method for reducing the storage required for these tuples to less than that required for representing said tuples individually.

In Bugajski, examples of these associative memories are given in Figure 2 (left and right grids) and Figure 3 (top grid), Note that the method shown uses a separate row for each association (tuple) in the sequence. Thus, this storage method represents all sequences as individual tuples. Thus, Bugajski does not teach any method for storing tuples in less space than the space required to store said tuples individually.

Since claim 1(b)(3) recites a storage-optimization method for an interior (non-terminal) node's tuples (associations), that can reduce the space required to less than that required to store individual tuples, and the cited prior art does not, it therefore does not disclose claim 1(b)(3).

Second, claim 1(b)(3) recites the exact form of an input-tuple sequence that can be compressed: Mutually-consecutive tuples. This term is defined precisely in the application (page 7, DESCRIPTION—Preferred Embodiment, 1. Tree structure, bottom of page).

In neither the referenced text of Bugajski, nor anywhere else in the patent is the concept of mutually-consecutive tuples defined or described. Nor is it demonstrated how such a sequence could be compressed.

Since claim 1(b)(3) recites a form of sequences that can be compressed, and Bugajski does not teach this form, Bugajski does not disclose claim 1(b)(3).

Claim 1(b)(4)(i) recites:

"recording the addition of a tuple that extends a tuple sequence by incrementing the length field of said tuple sequence"

This is said to be disclosed by Bugajski (col. 8, lines 11-13), which states:

"[...]process, and the unique data pair is added as an appendage to the output signal from that processor and it is passed upwardly to the highest level in the system. The appendages [...]"

Third, claim 1(b)(4)(i) recites extending a sequence of mutually-consecutive tuples by modifying the existing data structure of the sequence, which does not use additional storage, as described in the application (page 13, Operation, 3. Record Addition, (a)).

The appendage described in the cited text Bugajski (col. 8, lines 11-13), is used to transfer the storage of a data pair (tuple) to a higher level, after a level has been filled up (col. 8, lines 6-9). Thus the appendage cannot be put in the current level, and must be stored in a higher level, proving that the appendage requires additional storage.

Since claim 1(b)(4)(i) recites a method for adding a tuple which does not use additional storage, and Bugajski's appendage does require additional storage, the cited text does not disclose claim 1(b)(4)(i).

Fourth, claim 1(b)(4)(i) describes a method of adding a tuple within a single interior node.

Bugajski's appendage is used solely to pass tuples between levels, going to higher levels for storage, and to lower levels for reconstructing a record Bugajski (col. 8, lines 10-13, and lines 15-18).

Since the description of Bugajski's appendage does not teach a method for storing tuples within a single interior node, it does not disclose claim 1(b)(4)(i).

Fifth, claim 1(b)(4)(i) describes a distinct method for extending a sequence of mutually-consecutive tuples, which are precisely defined in the application (page 7, DESCRIPTION—Preferred Embodiment, 1. Tree structure, bottom of page).

The description of Bugajski's appendage does not teach a method for processing a sequence of mutually-consecutive tuples distinctly from other tuples, and indeed, does not even describe the concept of such a sequence, so it does not disclose claim 1(b)(4)(i).

Thus five independent reasons are given why the cited text does not anticipate claim 1. Since any one of these by itself would demonstrate novel material not anticipated by Bugajski, the Applicant respectfully requests that this claim be allowed.

3. Regarding on claim 3, Applicant respectfully traverses the rejection of claim 3 over Bugajski. Bugajski does not disclose each element of the claimed invention.

The office action states that: 1) claim 3(a) is disclosed by Bugajski (col. 9, lines 55-59), 2), claim 3(b) is disclosed by Bugajski (col. 9, lines 55-69), and 3) claim 3(c) is disclosed by Bugajski (col. 12, lines 55-67, and col. 13, lines 1-12).

The Applicant respectfully asserts that neither the cited text, nor any other description in Bugajski discloses claim 3(a).

Claim 3(a) recites:

"defining a gate field for one or more interior nodes"

Bugajski (col. 9, lines 55-59) teaches:

"each field value is associated with a numerical index value and stored in the dictionary. Each non-leaf or non-terminal node in the tree (such as 105, 108, etc.), however, leads to the creation of a table of associative memories whose two components are indexes to the memory tables of the node corresponding to the derivative branches or "children"

First, the gate field recited in claim 3(a) indicates which of a node's subtrees contain a field (leaf) from a given subset of the leaves (application, page 9, second paragraph).

The application's Figure 6, for example, shows a tree where gate-field values are shown for the interior nodes. The gate-field value tells which subtree(s) of the corresponding interior node lead to leaves in the subset, {A, D, F}. A value of 0 means neither subtree contains a leaf in said subset, 1 means only the left subtree contains a leaf in said subset, and 3 means both subtrees contain leaves in said subset.

The cited text, (Bugajski, col. 9, lines 55-59), does not teach any method for determining which of a node's subtrees contain a field (leaf) from a given subset. In fact, the non-leaf (interior) node's plurality of indexes described here only reference its immediate child nodes, (Bugajski, col. 9, lines 59-61, col. 15, lines 21-24, and col. 2, lines 64-66),.

In Bugajski's cited example, which describes Bugajski's Figure 1, the indexes from node 104, for example, only reference memory tables of nodes 105 and 106 (its two children). Thus the indexes described here cannot be used to indicate which of node 104's subtrees lead to leaves in a given subset.

Since the cited text does not provide the functionality of a gate field, it does not disclose claim 3(a).

Second, claim 3(a) recites a single gate field for an interior node.

The cited text describes a plurality of associative memories, components, and indexes in a non-terminal node. There are a plurality of the said dictionary numerical index values (Bugajski, col. 15, line 9-13 in a leaf node). Since there is a single gate field, and a plurality of the other structures, this proves that said other structures are different in nature and operation from a gate field, and do not disclose a gate field.

Third, for an interior node with 2 child nodes, a gate field requires 4 distinct values to represent all the combinations of child subtrees that may lead to leaves in the said leaf subset (Application, page 9, paragraph 4, and application Figure 6). Thus the number of distinct values of a gate field is fixed for a given interior node.

The tuples of indexes in Bugajski's non-terminal (interior) nodes must take on an arbitrarily high number of values to index into nodes at the next lower level (Bugajski, col. 3, lines 32-34, and col. 4, lines 39-44).

Since the said indexes used in Bugajski's interior nodes must take on an arbitrary number of values, they are different than a gate field, which takes on a fixed number of values. Therefore the cited text does not disclose claim 3(a).

Fourth, claim 3(a) states that gate fields are created for interior nodes.

The cited text states that "each field value is associated with a numerical index value and stored in the dictionary." Since said numerical index values are stored in dictionaries, said numerical index values are therefore only associated with leaf nodes, because only leaf nodes have dictionaries (Bugajski, col. 4, lines 37-39. Also Bugajski, col. 15, lines 16-19).

Since gate fields are only used in interior nodes, and the said numerical index values in the cited text are only used in leaf nodes, said numerical indexes are different from, and do not disclose the gate fields of claim 3(a).

Claim 3(b) states:

"setting the value of said gate field for each said interior node, to indicate which of said interior node's branches lead to leaf nodes in said subset"

Bugajski (col. 9, lines 55-69) states:

"each field value is associated with a numerical index value and stored in the dictionary. Each non-leaf or non-terminal node in the tree (such as 105, 108, etc.), however, leads to the creation of a table of associative memories whose two components are indexes to the memory tables of the node corresponding to the derivative branches or "children", whether terminal or non-terminal nodes. For example, a dictionary is created for each field represented by the terminal or leaf nodes such as serial number, model, year, plant codes, and so forth, whereas each non-terminal node leads to the creation of a table of associative memories whose [...]"

Fifth, as the Applicant has pointed out in the discussion of claim 3(a) (above), the cited text here, too, teaches no method with the functionality of a gate field.

Since the cited text does not teach the use of a gate field, it does not disclose claim 3(b).

Sixth, claim 3(b) recites a method for processing a subset of a tree's fields.

The cited text does not teach processing a subset of a tree's fields, and moreover does not mention or describe such a subset.

Since claim 3(b) recites a method for processing a subset of a tree's fields, and the cited text does not, the cited text does not disclose claim 3(b).

Claim 3(c) states:

"following paths that lead to said nodes"

Bugajski (col. 12, lines 55-67, and col. 13, lines 1-12) states:

"In order to directly interrogate the compressed structure just introduced, a query algorithm is used to according to an alternative aspect of this invention. Starting at the leaf nodes, and propagating up the tree, a list of memories which satisfy a particular query is collected at each node. A leaf node memory satisfies the query if its value is not excluded by the query and an associative memory satisfies the query if both of its component indexes refer to memories that satisfy the query. Referring to FIG 3, as an example, assume that one wishes to find all model X cars having Z brand tires. Starting at node A, field value XXX which represent model X cars, has an index value of 8, and ZZZ, at node B, is associated with an index value of 14. Now, at each node, the set of memories which satisfy the query is created whereas model X cars and Z brand tires have only one set which satisfies the query. Since all plant codes may be involved, for node B, the entire set is used. At node E, however, it is desired to know which of the plant codes have 8 for its left child, which would represent model X cars, but can have anything for the right child. Similarly, at node C, we wish to know of any indexes which contain 14 as the left child. Indexes 11 and 12 are relevant at node E, since the left child is 8 and the right child is irrelevant, and at node C, then, we are looking for any entry having a left child of 14 and a right child of 11 or 12. The only index which satisfies all of the above is the query set {113}."

Seventh, the algorithm described in the present invention searches a tree starting from an interior node (application, page 14, DESCRIPTION—Preferred Embodiment, 4, third paragraph from bottom), and following paths that lead down, toward the leaves (application, page 9, 3. Interior node gating, paragraph 2).

The algorithm described in the cited text starts searching at the leaf nodes and follows paths up the tree (Bugajski, col. 12, lines 57-58), toward an interior node (Bugajski, col. 13, lines 9-12, describes the search ending at node C; the root of the tree).

Since the two algorithms search the tree in opposite directions, they are different and the cited text does not disclose claim 3(c).

Thus seven independent reasons are given why the cited text does not anticipate claim 3. Since any one of these by itself would demonstrate novel material not anticipated by Bugajski, the Applicant respectfully requests that this claim be allowed.

4. Regarding on claim 20, the office action states that claim 20(b)(1) is disclosed by Bugajski (col. 8, lines 15-20).

Claim 20(b)(1) states:

"each leaf node is distinct from, and represents a subset of values from one of said dictionaries, and [...]"

Bugajski (col. 8, lines 15-20) states:

"Broadly, the method involves analyzing an incoming data stream in serial fashion, parsing the stream into short sequences of data elements, preferable pairs, and generating an output stream consisting of signals representative of each of the successive sequences of the input data. This output stream is provided to a second process or level which [...]"

Applicant respectfully asserts claim 20 teaches methods not anticipated by Bugajski.

Claim 20(b)(1) recites the representation of a subset of dictionary values.

The cited text does not teach a method for representing subsets of values from a dictionary corresponding to an input stream. It teaches the processing of the whole incoming data stream. Moreover Bugajski does not teach leaf nodes that represent a subset of dictionary values (Bugajski., col. 15, lines 16-19).

Since the application recites leaf nodes that are capable of representing a subset of dictionary values, and Bugajski does not teach either the representation of a subset of dictionary values, nor leaves that represent a subset of dictionary values, the cited text does not disclose claim 20(b)(1).

For clarification, compare application Figure 2, with Bugajski Figure 3. In Bugajski Figure 3, the nodes labeled A and B are leaves, and their contents are shown to be the indexes and values for an input stream. Said leaves are dictionaries.

In application, Figure 2, dictionaries are labeled (60), and correspond to Bugajski's dictionaries. The leaves in this Figure are labeled (61), and may represent a subset of dictionary values. Here, said leaves contain the counts of the corresponding dictionary values. Bugajski does not teach any equivalent construct.

Since the present invention teaches separate leaves and dictionaries and Bugajski doesn't, Bugajski does not disclose claim 20(b)(1).

In a previous office action, the Examiner stated that it would be obvious to have a plurality of trees index the same dictionaries. But this scheme would not work if two of said trees had different counts of same dictionary entry. This becomes a possibility when leaves are permitted to have subsets of their corresponding dictionaries.

While a dictionary could conceivably maintain a count for each entry, two different counts are required to represent the different numbers of instances in said two trees. What is required is maintaining separate counts of said dictionary entry in each tree. These required separate counts are essentially the leaves of claim 20(b)(1), (application, Figure 3 (61)).

This separation allows a second tree to share the same dictionary as the first tree, by maintaining a different set of counts in its leaves. Additional trees are recited in (application, page 5, Summary (b)), and (application, page 5, Objects and Advantages (b)(1-4)).

Since claim 20 recites a method of allowing a plurality of trees to share common dictionaries, possibly representing subsets of said dictionaries, and the cited prior art doesn't, it does not anticipate claim 20.

The Applicant has modified claim 20 to additionally recite a plurality of trees sharing one or more common dictionaries, to further clarify the difference between the present invention and Bugajski. This claim has been rewritten as dependent claims 5, 25, and 31.

5. Regarding on claim 22, the office action states that claim 22(b)(1) is disclosed by Bugajski (col. 9, lines 55-59, col. 12 lines 55-67, and col. 13 lines 1-12).

This is essentially the same thing that was said about claim 3, and the response is the same as for claim 3.

6. Regarding on claim 26, Applicant respectfully traverses the rejection of claim 26 over Bugajski. Bugajski does not disclose each element of the claimed invention.

The office action states that claim 26(b)(3) is disclosed by Bugajski (col. 2. lines 42-50), and that claim 26(c)(1-3) is disclosed by Bugajski (col. 9, lines 55-69, col. 12 lines 55-67, and col. 13 lines 1-12).

The Applicant respectfully asserts that neither the cited text, nor any other description in Bugajski anticipates claim 26.

Claim 26(b)(3) states:

"a gate field is defined for one or more interior nodes."

Bugajski (col. 2. lines 42-50) states:

"The present invention is directed toward the simultaneous compression of multiple streams of data into a dictionary or dictionaries, defined, for example, by the field structure of a database. However, the invention is not limited to such data structures, and a more general procedure is disclosed for directly searching the compressed body of data, and thereby accelerating the query thereof. Furthermore, the present invention is directed toward the compression of data having a plurality of field types, including means to accelerate a [...]"

First, claim 26(b)(3) recites the use of a gate field for interior nodes. Neither the cited text, nor anywhere else in Bugajski is said gate field taught. Since claim 26(b)(3) recites a gate field, and the prior art doesn't, it doesn't disclose said claim.

Second, the reference to Bugajski (col. 9, lines 55-69, col. 12 lines 55-67, and col. 13 lines 1-12) is essentially the same thing that was said about claim 3, and the response is the same as for claim 3.

Since claim 26 recites a gate field and the cited text doesn't, the Applicant respectfully requests that this claim be allowed.

Conclusion

The Applicant's claims demonstrate unique material and are patentable. Please reconsider this application.

8-22-05
Date

Respectfully submitted,
Alan Balkany

Alan Balkany
161 Commons Circle
Saline, MI 48176
(734) 944-7179
alankdkd@aol.com